Synthesis for Parameterized Systems

Mathieu Lehaut
with Béatrice Bérard, Benedikt Bollig, Tali Sznajder

Master class

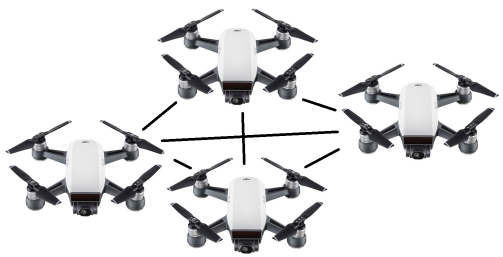
12 September 2019

The context

Distributed systems everywhere:

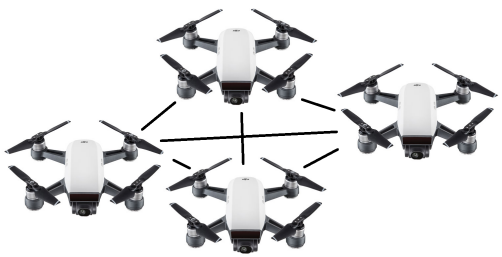
The context

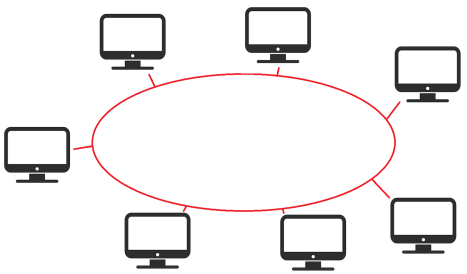
Distributed systems everywhere:



The context

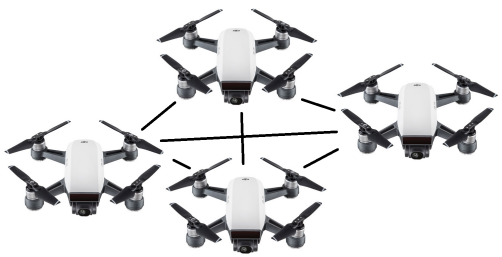
Distributed systems everywhere:

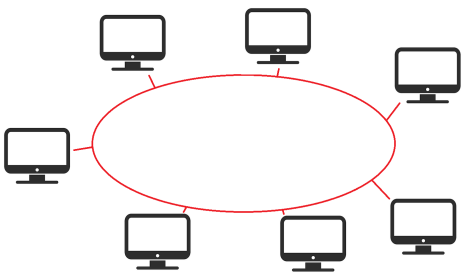




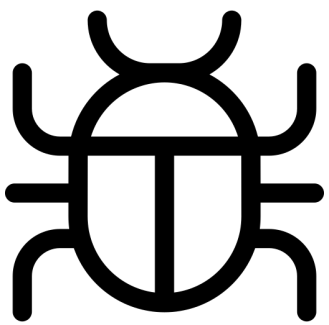
The context

Distributed systems everywhere:



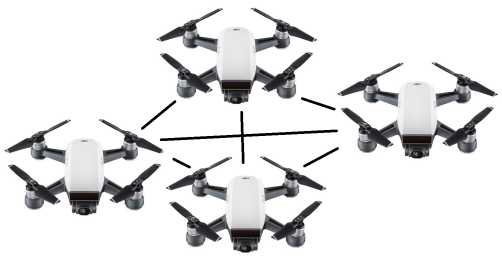


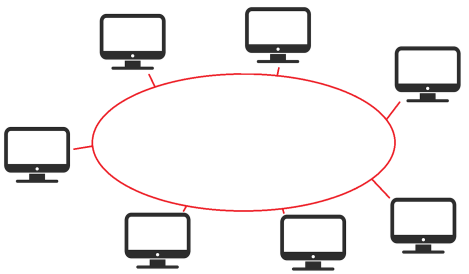
And bugs too.



The context

Distributed systems everywhere:





And bugs too.

Our focus

Parameterized Systems

A *parameterized system* is a distributed system with no fixed number of processes.

Our focus

Parameterized Systems

A *parameterized system* is a distributed system with no fixed number of processes.

Environment

Processes interact with an uncontrollable environment.

Our focus

Parameterized Systems

A *parameterized system* is a distributed system with no fixed number of processes.

Environment

Processes interact with an uncontrollable environment.

Specification

The system as a whole must complete some given task.

Synthesis

Synthesis problem

Given a specification φ, can we build a system S such that $(S||Env) \models \varphi$?

Synthesis

Synthesis problem

Given a specification φ, can we build a system S such that $(S||Env) \models \varphi$?

Motivations

- Bug-free by design
- Saving time

Problems

- Uncontrollable environment
- Unbounded number of processes

Behaviors of the system

Datawords

Words over $\Sigma \times \mathcal{D}$ with Σ a finite alphabet and $\mathcal{D}$ an infinite set.

Example over $\{req, ack\} \times \mathbb{N}$

$w = (req, 1)(req, 3)(ack, 1)(req, 6)(ack, 6)(ack, 3)$

Behaviors of the system

Datawords

Words over $\Sigma \times \mathcal{D}$ with Σ a finite alphabet and $\mathcal{D}$ an infinite set.

Example over $\{req, ack\} \times \mathbb{N}$

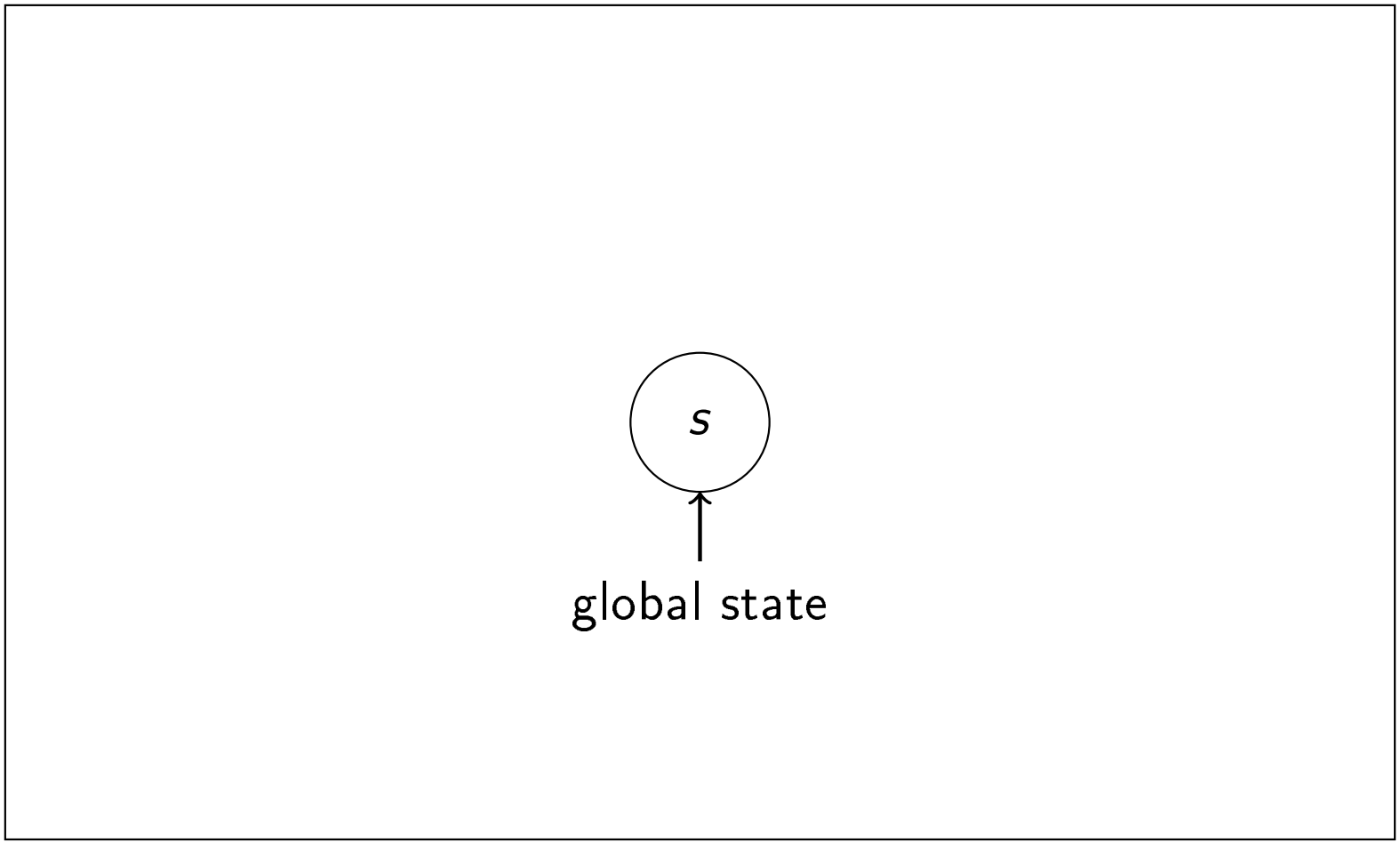
$w = (req, 1)(req, 3)(ack, 1)(req, 6)(ack, 6)(ack, 3)$

Specification

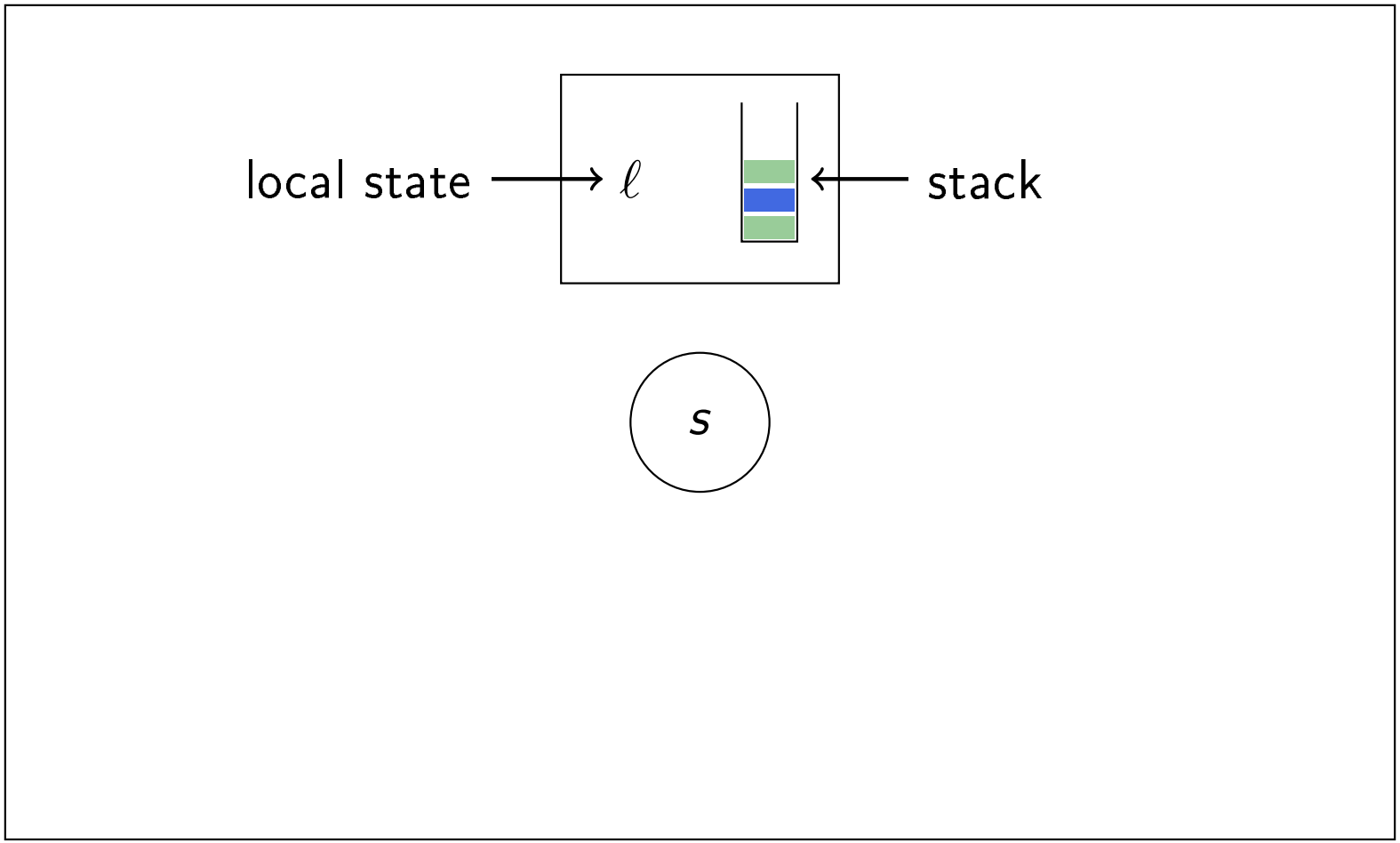
A specification is a set of *correct* datawords.

$\Rightarrow$ Specification as an automaton or a logic formula.

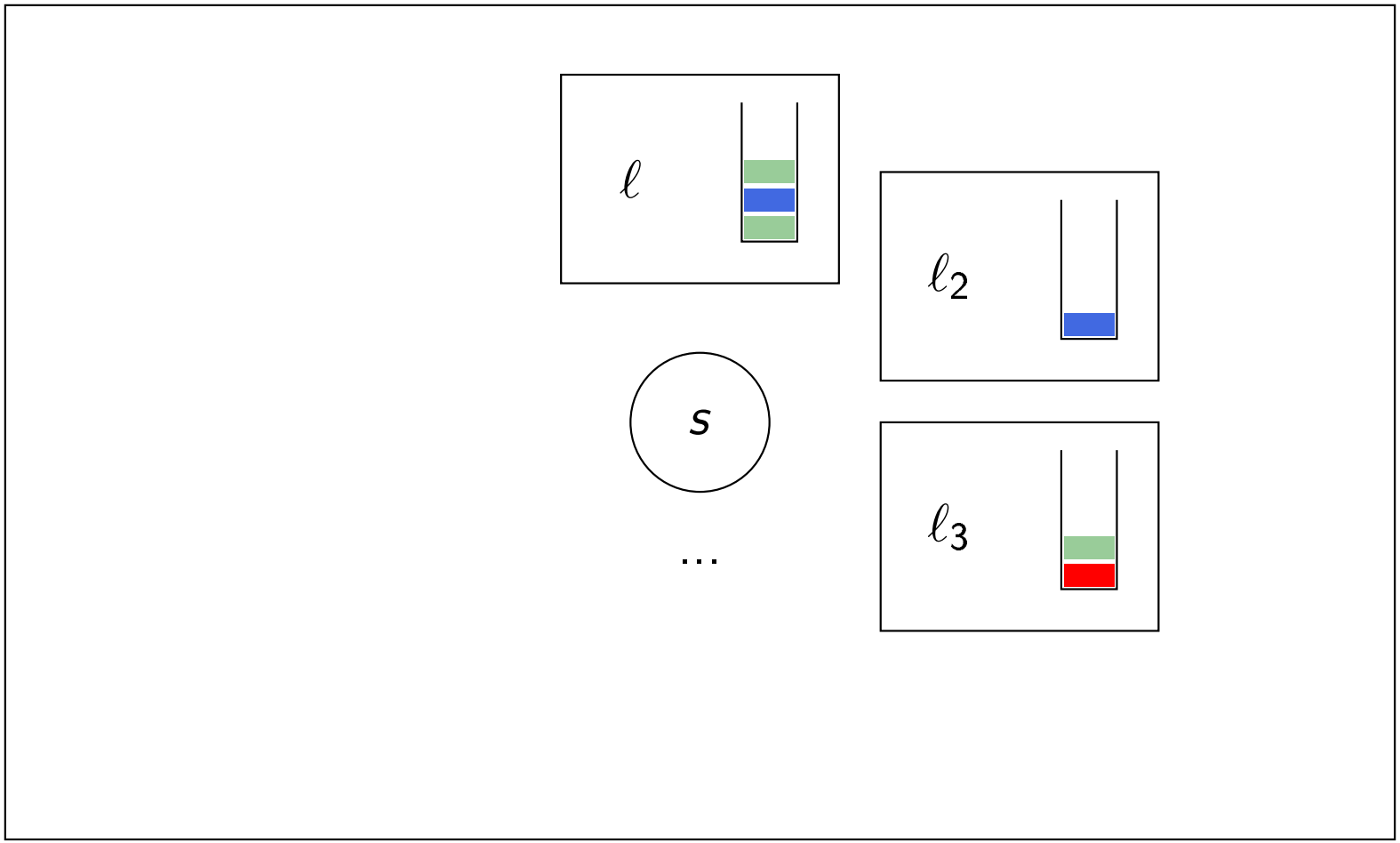
Parameterized Pushdown Systems



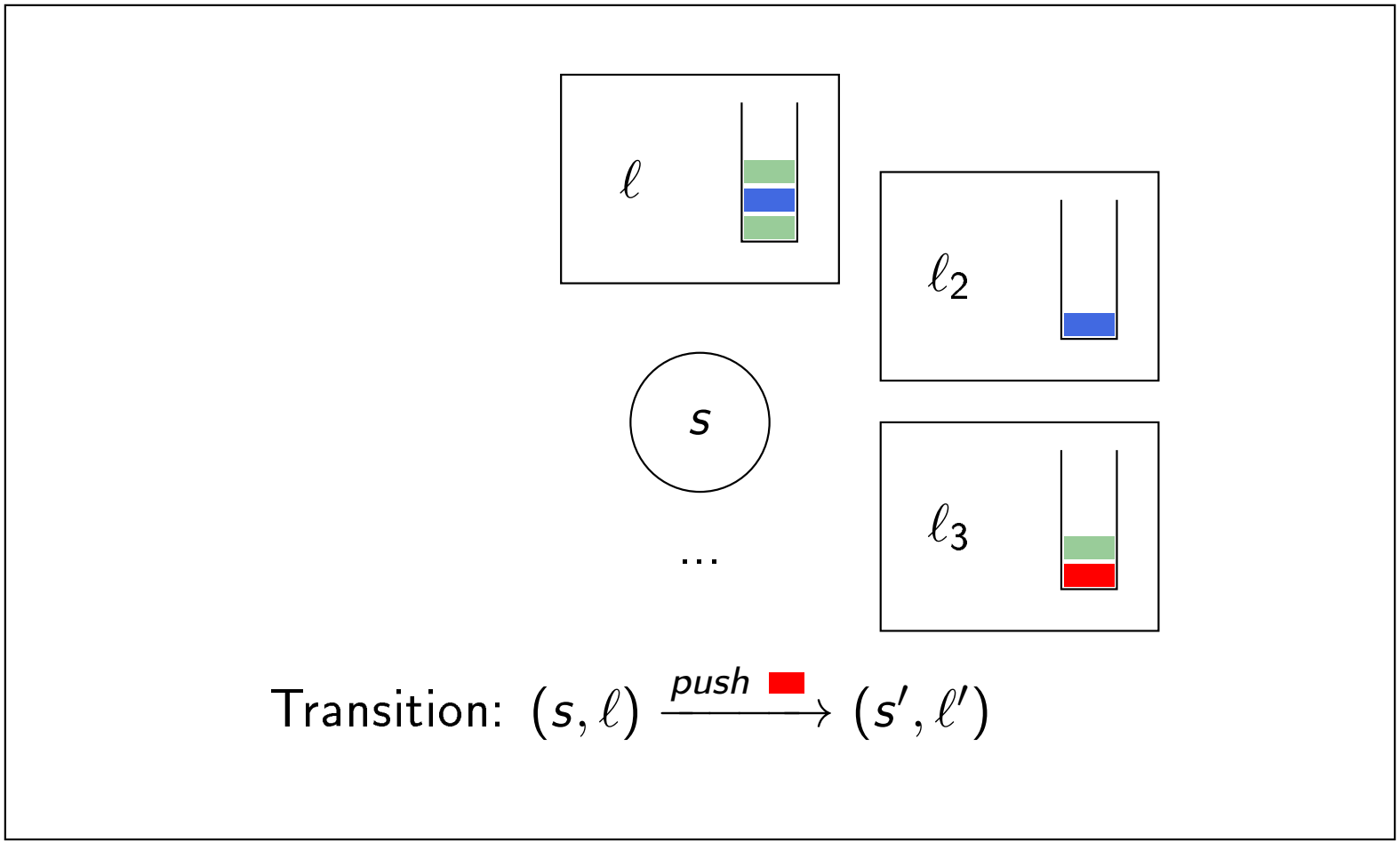
Parameterized Pushdown Systems



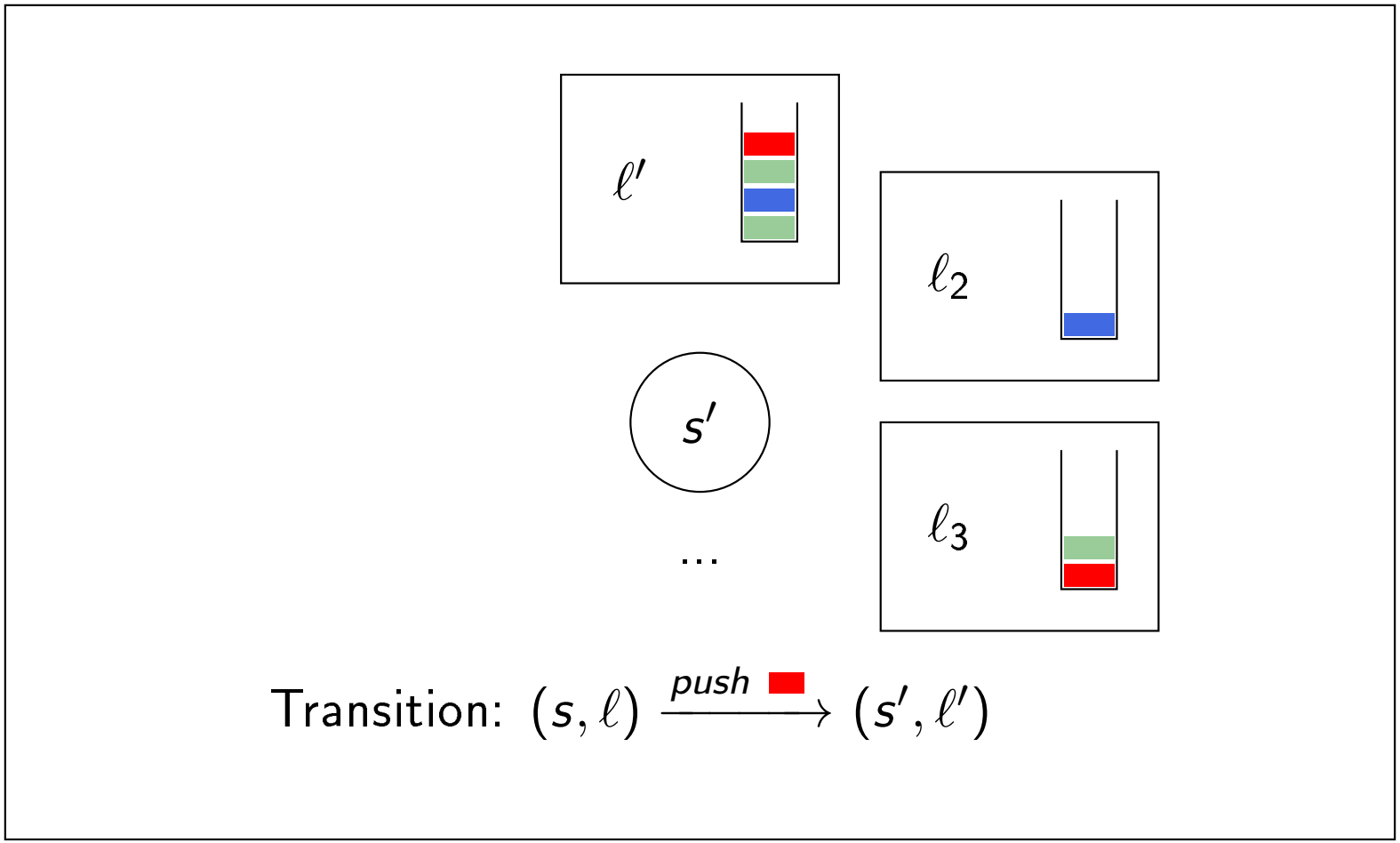
Parameterized Pushdown Systems



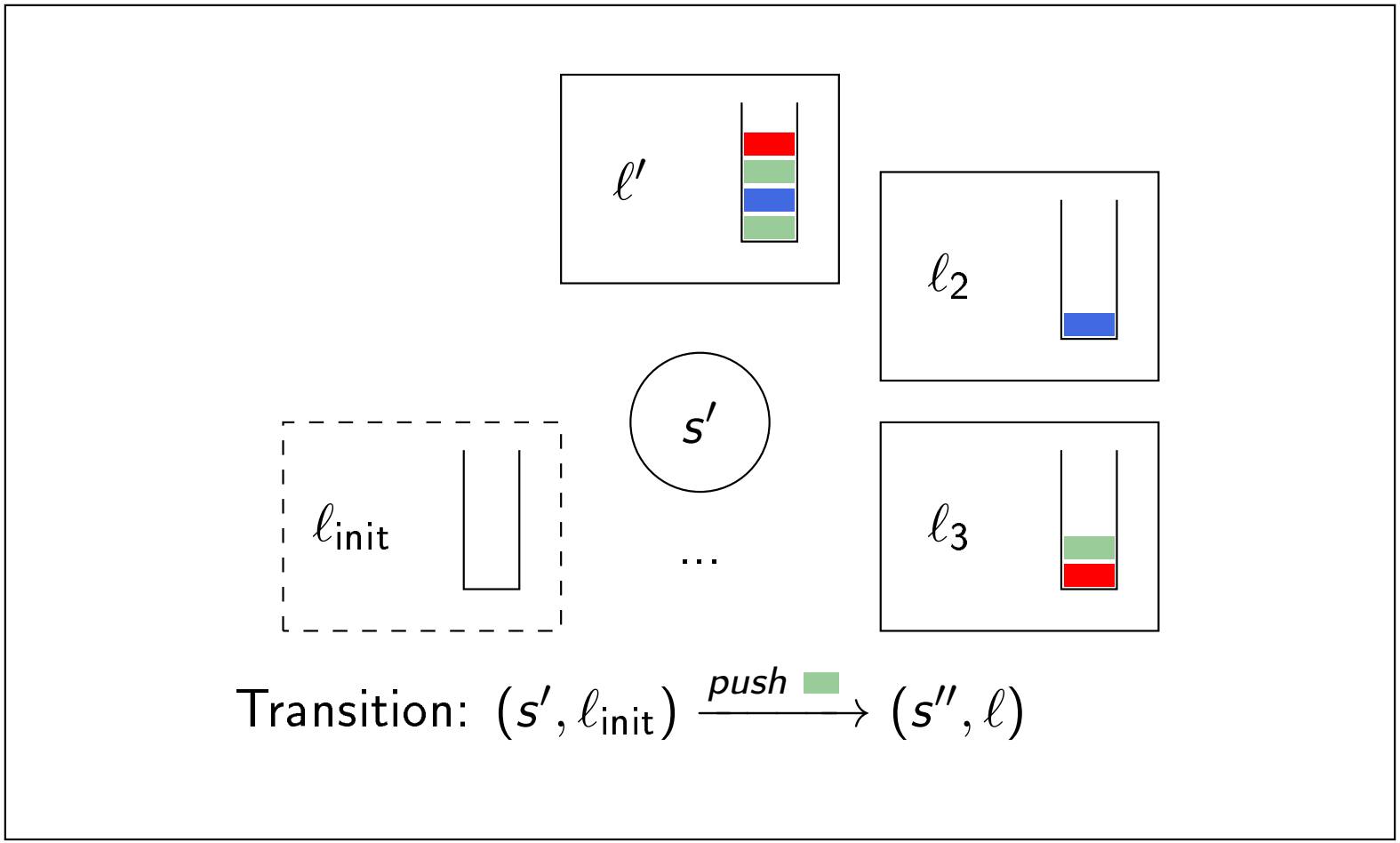
Parameterized Pushdown Systems



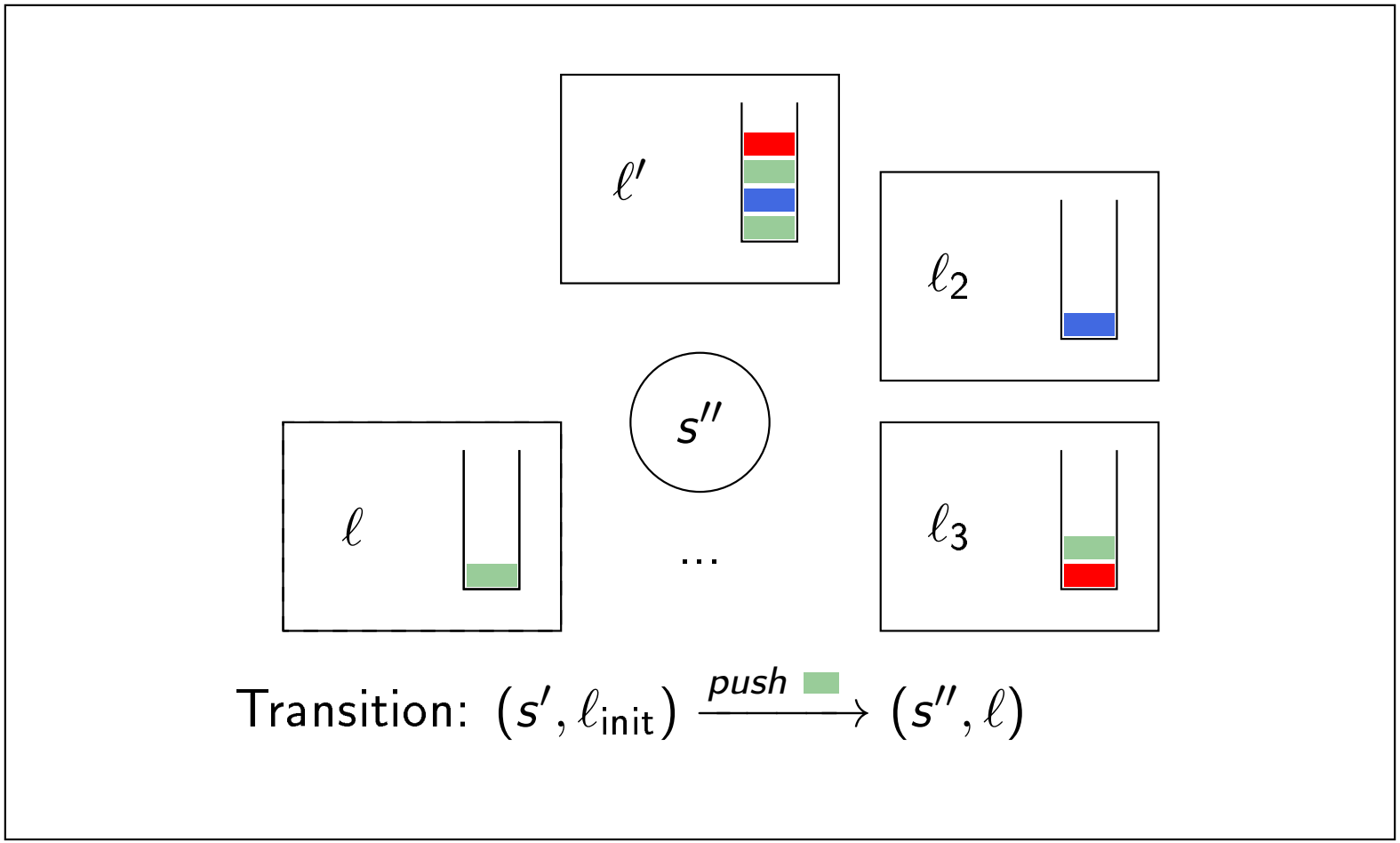
Parameterized Pushdown Systems



Parameterized Pushdown Systems



Parameterized Pushdown Systems



Results

Need to restrict behaviors for decidability!

Results

Need to restrict behaviors for decidability!

Restriction: round-bounded behaviors [La Torre et al., 2010]

1 round:

- ✓ 1 1 1 1 2 2 2 3 3 3 3 3 4 5 5 5 ...
- ✓ 1 1 1 1 2 2 2 2 2 5 5 5 8 8 8 ...
- X 1 1 1 2 2 3 3 1 4 4 6 ...

Results

Need to restrict behaviors for decidability!

Restriction: round-bounded behaviors [La Torre et al., 2010]

1 round:

✓ 1 1 1 1 2 2 2 3 3 3 3 3 4 5 5 5 ...

✓ 1 1 1 1 2 2 2 2 2 5 5 5 8 8 8 ...

X 1 1 1 2 2 3 3 1 4 4 6 ...

N rounds:

1 1 1 2 3 4 4 | 1 3 3 4 5 | 3 3 5 6 6 6 7 | ...

Results

Need to restrict behaviors for decidability!

Restriction: round-bounded behaviors [La Torre et al., 2010]

1 round:

✓ 1 1 1 1 2 2 2 3 3 3 3 3 4 5 5 5 ...

✓ 1 1 1 1 2 2 2 2 2 5 5 5 8 8 8 ...

X 1 1 1 2 2 3 3 1 4 4 6 ...

N rounds:

1 1 1 2 3 4 4 | 1 3 3 4 5 | 3 3 5 6 6 6 7 | ...

Our results

- Emptiness problem is PSPACE-complete
- Synthesis problem is decidable (inherently non-elementary)

First Order Logic

Let Σ be a finite alphabet.

FO

First order formula:

$$\varphi ::= x = y \mid a(x) \mid \neg\varphi \mid \varphi \vee \varphi' \mid \varphi \wedge \varphi' \mid \forall x.\varphi \mid \exists x.\varphi \mid$$
$$x \sim y \mid x < y \mid x = y + 1$$

with $x, y, \ldots$ variables and $a, \ldots \in \Sigma$

First Order Logic

Let Σ be a finite alphabet.

FO

First order formula:

$$\varphi ::= x = y \mid a(x) \mid \neg\varphi \mid \varphi \vee \varphi' \mid \varphi \wedge \varphi' \mid \forall x.\varphi \mid \exists x.\varphi \mid$$
$$x \sim y \mid x < y \mid x = y + 1$$

with $x, y, \ldots$ variables and $a, \ldots \in \Sigma$

Example

$\varphi = \forall x.(req(x) \Rightarrow \exists y.(y \sim x \wedge y > x \wedge ack(y)))$

Synthesis problem and results

$$\Sigma = \Sigma_{sys} \uplus \Sigma_{env}$$

Synthesis problem viewed as a game

Given a FO formula φ, is there a winning strategy for the System?

Synthesis problem and results

$$\Sigma = \Sigma_{sys} \uplus \Sigma_{env}$$

Synthesis problem viewed as a game

Given a FO formula φ, is there a winning strategy for the System?

Our results (WIP)

- $\mathrm{FO}^2(\sim, <, +1)$ is undecidable
- $\mathrm{FO}(\sim)$ is undecidable unless processes are partitioned into System and Environment processes (without overlap)

Synthesis problem and results

$$\Sigma = \Sigma_{sys} \uplus \Sigma_{env}$$

Synthesis problem viewed as a game

Given a FO formula φ, is there a winning strategy for the System?

Our results (WIP)

- $\mathrm{FO}^2(\sim, <, +1)$ is undecidable
- $\mathrm{FO}(\sim)$ is undecidable unless processes are partitioned into System and Environment processes (without overlap)

Thank you for your attention!